

Alert-driven E-Service Management⁺

Dickson K.W. Chiu¹, Benny W. C. Kwok¹, Ray L. S. Wong¹, S.C. Cheung², Eleanna Kafeza³

¹Department of Computer Science and Engineering, The Chinese University of Hong Kong

²Department of Computer Science, Hong Kong University of Science and Technology

³Department of Marketing and Communications, Athens University of Economics and Business

email: dicksonchiu@ieee.org, { bennykok2000, digital_ray }@yahoo.com, scc@cs.ust.hk, kafeza@aueb.gr,

Abstract

Process management technology has recently been employed not only within businesses but also in provision of E-services over the Internet. Urgent requests and critical messages in these systems (referred to as alerts) should be delivered and handled timely. In particular, humans are often involved in critical conditions. Presently, most E-service systems cannot address these requirements, and alerts are often handled in an ad-hoc manner. In this paper, we propose a sophisticated alert management system for effective e-service integration under urgent constraints. We develop a model for specifying alerts, in which alerts are associated with service requests and a set of parameters are captured for their routing. The alert monitor matches the service provider to receive an alert, based on the alert specification. We then propose a routing mechanism that is initiated when the alert message is not acknowledged or serviced within the deadline, so that the alert can be forwarded to other suitable services if necessary. Monitoring is especially essential to ensure timeliness and availability of services, otherwise suitable exceptions should be raised and handled. We outline our implementation framework of an Alert Management System (AMS) which includes Web Services for B2B interactions, together with multiple-platform support for human users. We demonstrate our approach with an example medical house-call AMS.

1. Introduction

Recent advances in Internet technologies have created a global platform for organizations and individuals to communicate among one another, carry out various commercial activities, and provide value-added services. E-services refer to these services provided over the Internet. In the current highly competitive and dynamic digital economy, E-services need to respond actively and timely

to customers' needs. For example in a B2C scenario, when a product recommendation portal finds a suitable product for a customer, the customer should be *alerted* promptly with the search results because second-hand items and clearance stock might be sold out quickly. Similarly, business-to-business (B2B) interactions may also have such requirements. Timely communication of accurate information is a key success factor for the provision of quality E-services. In particular, for E-services with stringent urgency requirements, such as healthcare and security applications, E-service providers must respond actively and very timely to requestors' needs as this may be crucial to success or failure.

There are several issues to be considered in this problem. The alert model should include various alert types and parameters that qualify the E-service provider to receive an alert. Apart from service suitability, application specific considerations like costs, waiting time, service time, etc., may also be important. In addition, routing, monitoring, and logging the alerts are also mandatory functionalities, in order to shift the burden of these communications from the manual work to an automated system. To take advantage of the connected Internet environment, we propose an alert management system (AMS) to address the problem of urgent support to E-services. The AMS aims to minimize delays by providing a monitoring system. This paper generalizes and extends our previous work on workflow modeling [6], process integration [4], and contract enforcement [5]. The contributions of this paper are as follows: (i) a conceptual model for specifying alerts based on the requirements of cross-organizational processes and a set of routing parameters; (ii) a practical architecture for the AMS based on contemporary Web Services for B2B interactions, together with multiple-platform support for human users; (iii) an algorithm for matching service providers to alert requirements; (iv) a mechanism for (re-)routing alerts and increasing their urgency when alerts are not acknowledged or processed within deadline; and (v) demonstrate the applicability of a

⁺ This work was supported by the Hong Kong Research Grant Council with an Earmarked Research Grant (HKUST 6170/03E).

AMS in a medical house-call system prototype. The rest of our paper is organized as follows. Section 2 discusses a motivating example drawn from a medical house-call application to demonstrate our approach throughout this paper. Section 3 describes our alert conceptual model that consists of two parts, namely, the processes and the alerts. In section 4, we present our architecture for an AMS and the mechanisms for monitoring and routing the alerts. Section 5 discusses some implementation details. We conclude in section 6 with our future work.

2. Motivating Example and Related Work

In Hong Kong, there are some medical corporations providing on-call service called “House Call”. Patients can call the medical corporations and request a doctor consultation at their home. When the call center of the medical corporation receives a call from a patient (either electronically or by phone), they will arrange a doctor to perform the consultation immediately or to make an appointment for the consultation in patient’s requested time. The call center should find a doctor with the required specialties (if any) from a list of off-duty doctors first, then from a list of on-duty doctors, and lastly, from a list of doctors from medical partners. In some cases, the system may also need to find a nurse to assist doctors to carry out the consultation. When the required personnel are found, the patient will be confirmed. At the same time, the patient’s medical records may be required to send from hospitals and other clinics to the doctor’s mobile device. After the consultation is finished, the doctor submits a medical report for the consultation. Lastly, the patient or his/her insurance company is charged for the consultation. Figure 1 depicts a summary of the user requirements in Unified Modeling Language (UML [17]) Use Case Diagram..

With UML Activity Diagrams, Figure 2 summarizes some of the important processes of the medical house-call

system, in which the concepts of alerts play an important role. Examples include: (a) finding a doctor for the patient; (b) confirming the doctor and patient on the house-call; and (c) the exception handling procedure of the doctor’s absence from a call, respectively. These motivate our generalized alert conceptual design and algorithms.

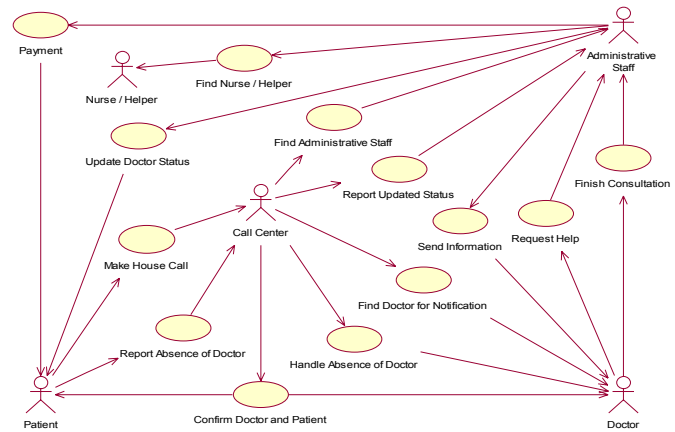


Figure 1. A use case diagram in UML for the example medical house-call system

Before automation, manual procedures have many problems in providing such services. Call receivers and administration staff of the call center require a lot of knowledge and experience in order to handle it correctly and timely. There is a strong need for systematically handling it by capturing their knowledge and experience should be captured by an E-service system. In particular, the process is error prone and there are many possible exceptional cases, such as, failure of finding suitable personnel, absence and lateness of the personnel, etc.

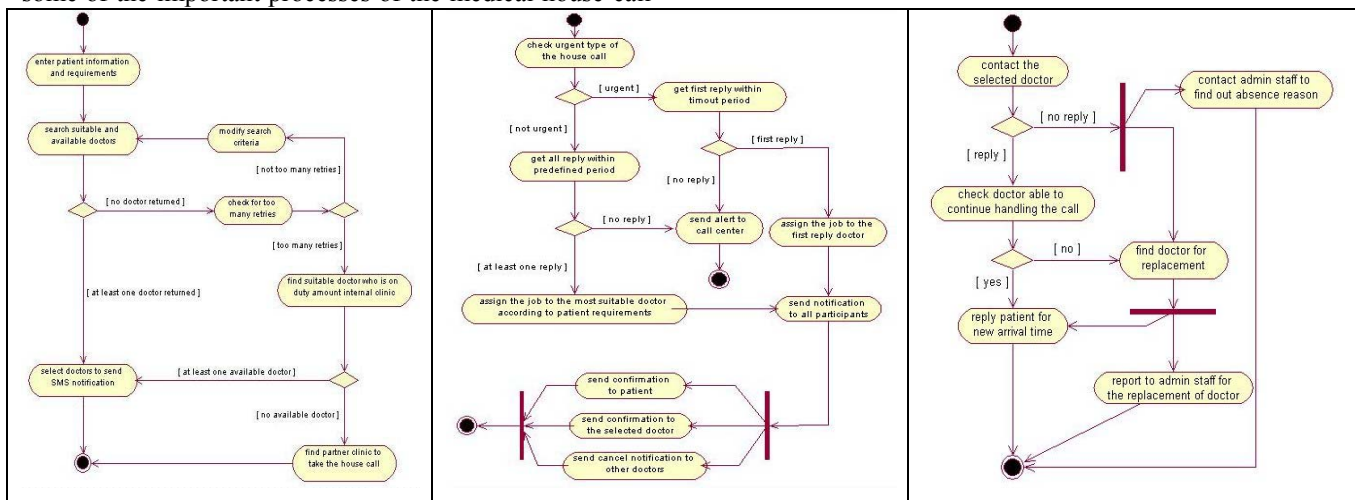


Figure 2. UML Activity Diagrams for (a) finding a doctor, (b) confirming doctor and patient, (c) absence of doctor, respectively

The call center is also a bottle neck part in the whole E-service process. It is difficult to receive too many requests at the same time manually. If it is automated as E-services, user can request house call via the Internet through different devices. Besides through a browser on a PC, users may call through other mobile devices. On the other hand, patients with long-term sickness can also call via pre-programmed devices with a simple interface (such as just an electronic button). In the manual system, service personnel can only receive alert message by phone or pager. The new system allows them to receive from or reply to the system via different channels, such as, Short Message Service (SMS), email, fax, Personal Digital Assistance (PDA), I seek you (ICQ), etc. Costly and inefficient manual calls and retry calls can be mostly eliminated and therefore this brings a great benefit to the medical corporation.

Communication with medical partners and hospitals is a time-consuming process in the existing system. If their systems are also computerized in such a way, they can form a service grid with a common interface, streamlining E-service provision among partners. In this paper, we shall concentrate on our new proposals of alert mechanisms, while other details on inter-organizational process interoperations can be found in our earlier work [4][5][8].

In the context of workflow management systems (WFMS), we have recently proposed to separate user alerts from user sessions with the WFMS to improve the flexibility in our ME-ADOME system [2]. Online users are alerted through ICQ, with the task summary and reply URL as the message content. If the user is not online or does not reply within a pre-defined period, the WFMS will send the alert by email. At the same time, another alert may be sent via SMS to the user's mobile phone. Whatever the alert channel has been, the user need not connect to WFMS on the same device, or even on the same platform. For example, after receiving a SMS alert, the user may use his handset to connect to the WFMS via Wireless Application Protocol (WAP), or he/she may reply with an SMS message. Alternatively, the user may find a PC with Internet connection or use his/her PDA to connect to the WFMS. To our knowledge, there has been no other WFMS employing this approach. So far, there has been no detailed study on the linkage between alerts and workflows, neither in medical applications or e-services.

Raghupathi et al. [21] point out that information technology (IT) is important to healthcare and the estimated IT expenditure on healthcare in 2002 is 21.6 billions in the United States. New healthcare applications supporting IT-based strategy are required for meeting competitive challenges. Ammenwerth et al. [1] also report that one of the major issues that mobile technologies can help in hospitals is communication and reachability management. Communications that include the patient, the message sender, and the urgency are useful. Hripcsak et al. [13] preliminarily

identify the need for event monitors, and describe some of the requirements of such monitors, such as, tracking medical events, looking for clinically important situations, and sending messages to the providers. Eisenstadt et al. [9] further categorize messages as *alerts*, *results*, and *replies*. The limitation of their approach is that they only focus on alerts that can be handled by 2-way pagers. Ride et al. [20] argue that the problem of figuring out to whom the message should be sent is a difficult one. They only suggested some ad hoc solutions, e.g., sending a message to whoever has recently examined the patient electronic record. In summary, the study of alerts even in healthcare informatics, which involves life critical application, is still preliminary. This also motivates us to conduct an in-depth study on alerts for further applications in these areas.

3. Alert Conceptual Model

E-Service Application Logic (J2EE/WFMS/...)
Alerts (AMS)
Events / Exceptions (Web Services)

Figure 3. The role of alerts in an information system

Figure 3 further clarifies the role of alerts in an information system. Alerts capture urgency requirements of a task, as required by some E-service application logic (which can be capture in any type of software implementations). An alert can either be (i) *asynchronously* received by external events or exceptions, typically incoming E-service requests, or (ii) *synchronously* generated by internal E-service application logics. It should also be noted that exceptions are subclasses of events [6]. However, different from general events, alerts have much more specific attributes, in particular, urgency and service requirements. Different from exceptions, they need not relate to abnormal behaviors. Alerts received or generated have to be *handled* by the AMS by requesting services from either (i) internal information systems, (ii) human service provider, or (iii) external E-service providers. As such, the AMS also handles inter-organizational interactions.

As summarized in Figure 4, our *alert conceptual model* consists of two parts, the process and the alerts model. This is because besides an alert itself, we must also consider the requirements of the tasks associated with the alert. The process model describes a process that abstracts the procedures of an organization. As an extension to existing process models such as [23], our process model abstracts information regarding roles and their schedules of persons or E-service providers possessing these roles. A preliminary version of our process model in the context of WFMS is available in [6]. We further incorporate alerts and their routing in this paper.

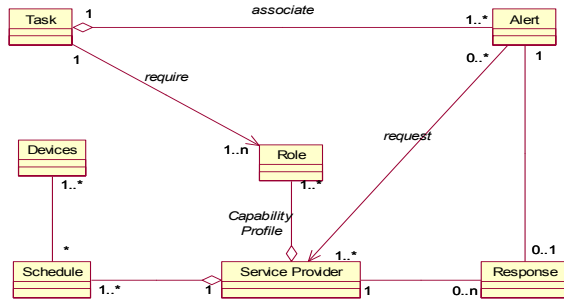


Figure 4. UML Class Diagram for alert concept module

3.1. The Process Model

In an organization, some of its tasks have to be performed by another organization via E-services. We call them *communication tasks*. Besides the task name and description, the requirements for performing each of them has to be specified as well. We call such requirements *service-provider roles*. Communication tasks generate *alerts* that are routed to matching service provider(s), which can be human or electronic (E-service provider). This task has to be carried out by a service provider playing the required role that reflects the requestor's requirement. Sometimes, an alert might need to be sent to several service providers (e.g., doctors, or medical partners) with the similar capabilities. Thus, the three kinds of elements in a process of AMS, viz., *Task*, *Capability Profile*, and *Schedule*, as illustrated as follows:

1. *Task* tuple definition: $(T_{id}, T_{des}, \{R_{id}\})$

T_{id}	Unique identifier of the task
T_{des}	Task description
$\{R_{id}\}$	Set of roles required to accomplish the task

2. *Capability Profile* tuple definition: $(E_{id}, \{R_{id}\})$

E_{id}	Unique identifier of a service provider
$\{R_{id}\}$	Set of roles that the service provider plays

3. *Schedule* tuple definition: $(E_{id}, \{A_b, T_{slot}, \{D_{id}\}\})$. For E-service providers, they are usually always available and their contact channel is usually web services.

E_{id}	Employee who has the schedule
A_b	Availability of service provider during the time-slot
T_{slot}	Time-slot for the service provider (included date and time period)
$\{D_{id}\}$	Set of available contact channels (e.g. mobile phone, fax, PDA, computer, pager, telephone, Web Services, etc)

3.2. The Alerts

During the execution of an urgent process, an alert has to be sent to the matched service provider(s) that are avail-

able for service. Besides identification information, an alert has an urgency level and a response flag that indicates whether it has to be acknowledged by a deadline. The urgency of the alert U could be a function of time. Normally when not acknowledged, the AMS increases alert urgency with time, as further explained in the next section. Although traditionally alerts are small messages, we generalize the concept of alert to include any additional information are urgent, important, or that can better justify the request. For example, an audio file or an image could describe the injured patient better than text. Note that there can be one or more alerts associated to a communication task. *Association* models the relationship between a task and a set of alerts, i.e., a task will trigger at least one alert. Response is user reply of a alert sent by the AMS. If the alert message is set response flag to be true, it means reply from the receiver of the alert is a must.

After receiving an alert, the service provider has to check it with its local policies and respond accordingly. On the other hand, the AMS needs to deal with those alerts that are not acknowledged by the deadline. From reported practice, we observe that as long as an alert is not acknowledged, the service provider assigned to handle the alert may change. As a result, the AMS can revise the matching between the alert and the service provider. In addition, as the time passes without any acknowledgement, the urgency of the alert should increase as well. Thus, this may in turn change the service provider to be requested. In our model, we propose a flexible approach, in which a strategy can be defined by the administrator.

Each service provider has either communicated to all interested parties a list of the service descriptions that it provides and/or it allows for a search on the provided web-services for example where the alert description tries to match the existing services and/or has published the services though a broker. Thus, the four kinds of elements concerning the alerts of AMS, viz., *Alert*, *Association*, *Response* and *Service Providers*, are illustrated as follows:

1. Alert tuple definition: $(A_{id}, A_{sub}, A_{ct}, \{\{E_{id}\}, \{R_{id}\}\}, U_{rg}, R_{sp}, T_{dl})$

A_{id}	Unique Identifier of the alert
A_{sub}	Alert subject
A_{ct}	Alert content (e.g. text message, patient's record, X-ray image, etc.)
$\{\{E_{id}\}, \{R_{id}\}\}$	Receivers of the message (a group of employees and a set of roles)
U_{rg}	Urgency of the message
R_{sp}	Necessary of response for the message (Boolean value)
T_{dl}	Deadline of the response

2. *Association* tuple definition: $(T_{id}, \{A_{id}\})$

T_{id}	Unique identifier of the task
$\{A_{id}\}$	A set of alert triggered by the task

3. Response tuple definition: (Eid, Aid, Rmsg, Tstp)

E _{id}	Receiver (Service Provider ID)
A _{id}	Alert which the response belongs to
R _{msg}	Response message
T _{stp}	Timestamp indicated the time when the alert was received

4. Service Provider tuple definition: (d_{id}, D_{di}, {(C_{di}, P_{di})}, Q)

d _{id}	Identification information of the service provider
D _{di}	Description information about the service provider
{(C _{di} , P _{di})}	Set of pairs: C _{di} describes the service that the service provider offers, P _{di} indicates the average response time
Q	Queue in which the set of alerts are waiting to be served by the corresponding service provider.

4. Alert Management System

In this section, we first present our system architecture for a flexible AMS, which supports sophisticated alert monitoring and routing. We then present an example service provider matching algorithm. In subsection 4.3, we deal with the problem of unacknowledged alerts with an urgency strategy definition schema and raising urgencies. In subsection 4.4, we present our overall monitoring framework for an AMS. Finally, we outline the Web Services definition required by the AMS.

4.1. AMS Architecture

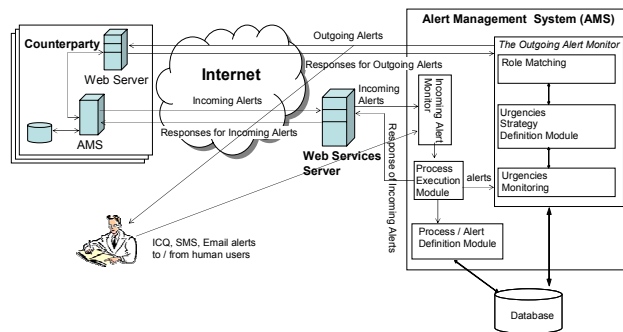


Figure 5. Alert Management System architecture

Our AMS architecture supports an organization to be both a service provider and requester at the same time. Each organization can use the AMS to both receive and submit alerts. The AMS consists of two major parts (cf. Figure 5). The *Incoming Alert Monitor* is responsible for receiving and queuing alerts and enacting the corresponding services (processes). Incoming alerts are received as requests to a Web service, where responses can be made

by the *Process Execution Module*. The *Outgoing Alert Monitor* subsystem is responsible for creating and submitting the alerts by means of Web services requests to the corresponding service providers, and monitoring their responses. The *Outgoing Alert Monitor* subsystem consists of three modules: the *Urgencies Strategy Definition*, the *Role Matching* and the *Urgencies Monitoring* modules. The *Urgencies Strategy Definition* module specifies the policies that will be followed if the alert is not acknowledged within the deadline. The *Role Matching* module is responsible for identifying the role to which the alert will be forwarded. The *Urgencies Monitoring* module is responsible for applying the strategies defined at the urgencies strategy definition. In addition, the *Process and Alert Definition* module supports a tool with which users may define the tasks and their associated alerts according to the AMS model. As for human users, they are communicated with ICQ, SMS, and email too. As such, a hospital supporting only manual record retrieval may still participate in the integration. The alert can be routed to a clerk, who input manually the required response to the requestor's Web service.

4.2. The Service Provider Matching Module

Input: alert requests

Output: (e_{id}, d_{id})

Service Provider Matching

For every alert (α_{id}, E, D_a, {R_i}, U, B, T)

// find the service provider that can play the required roles

Compute the set $SP = \{ (d_{id}, C_i) : \text{such that } \forall (d_{id}, \{R_i\}) \{R_{ij}\} = \{R_j\} \text{ and } \forall (d_{id}, D_{di}, \{(C_{di}, P_{di})\}, Q) C_i \text{ is the web service to play the roles} \}$

// if no such service provider exists, find another playing a superset of roles

If $SP = \emptyset$ {

 Increase urgency

 Upgrade alert (α_{id}, E, D_a, {R_i}, Urgency for service provider role substitution, B, T)

 exit

}

// find the service provider that are available to receive the alert before its deadline

Compute the set $SP_{sub} = \{ (d_{id}, C_i) : (d_{id}, C_i) \in SP \text{ such that expected submission time} + \text{average response time} (P_i) \leq \text{deadline} \}$

// if no such service provider exists, find another playing a superset of roles

If $SP_{sub} = \emptyset$ {

 Increase urgency

 Upgrade alert (α_{id}, E, D_a, {R_i}, Urgency for service provider role substitution, B, T)

 exit

}

Select the first service provider of SP_{sub} to submit the alert.

Figure 6. Service provider matching algorithm

The service provider matching module is responsible for searching a service provider for each alert. The service provider matching algorithm (cf. Figure 6) searches for those service providers that can play the role required for the alert. The algorithm then selects those that have a response time that is less than the deadline. This further re-

stricts the set of service providers that can receive the alert. If the matching is successful, one service provider is selected. In case no matching is available (i.e., there exists no service provider with the requested role that can meet the deadline), the algorithm upgrades the alert by expanding the roles whenever possible. After the matching, the *active alerts table* keeps all instantiated alerts and whether the alert has been acknowledged or not. A typical entry of the active alerts table is shown as follows, where 2/5: 17:00 is the current time of the system, the alert has been propagated, and there has been no response:

Table 1: Active Alerts Table

Active alerts table : Alert	Resp.
((002, Hospital A, phone: 9745678, Doctor Y, Urgent, Response, 2/5: 17:00)	∅

4.3. The Urgencies Strategy Definition Module

The urgencies strategy definition module is a tool for defining the policies according to which the urgencies of the alert will evolve. Moreover, this module is responsible for keeping and updating status information for the alerts. In our alert model, every alert is associated with an urgency value and a deadline, while every service provider associates an average response time for every service that it provides.

During the specification phase, the administrator has to specify the *urgencies strategy tables*. If an alert is not acknowledged, the AMS raises its urgency. An *urgencies strategy table* defines the policies for every urgency increase and the additional actions that should be taken. The administrator may define different urgency strategy tables for different types of alerts. For example, we could define the urgency values from the ordered set {Low, Normal, Urgent, Very Urgent, Critical, Very Critical} and a default *urgency function* as follows:

$$U_{002}(t) = \begin{cases} \text{Urgent} & t \leq T \text{ (default)} \\ \text{Very Urgent} & T < t \leq T + dt_1 \\ \text{Critical} & T + dt_1 < t \leq T + dt_1 + dt_2 \\ \text{Very Critical} & T + dt_1 + dt_2 < t \leq T + dt_1 + dt_2 + dt_3 \end{cases}$$

Table 2. Example urgencies strategy table

Urgency002	Action
Urgent	default
Very Urgent	Submit a second alert to the same SP, notifying about the approaching deadline
Critical	Redirect the alert to another SP that has the best response time
Very Critical	Send the alert to several SPs and accept the results of the one that response first

Table 2 shows an example urgencies strategy table. Here, let us consider the association of alert 002 of Table 1 with this table. The default level for alert 002 is *Urgent*. When the priority increases to *Very Urgent* because there has been no response, the AMS creates another alert message to notify the service provider about the eminent deadline. If still there is no response, the AMS tries another service provider with the same capabilities and the best response time. If this step also fails, the priority further increases to *Very Critical*, where all available service providers with requested capabilities will receive the alert.

4.4. The Service Provider Monitoring Module

The service provider monitoring module is responsible for sending alert messages, receiving acknowledgments, maintaining alert status, and logging information. For every acknowledgment message received, the service provider monitoring module updates the status information of the associated alert. It tags that the alert has been “taken care of”. If the alert message has been sent to several service providers, the first one to acknowledge is assigned to the task and then reconfirmed. Other service providers will receive a cancellation message instead. Then for every alert of the active table with its deadline expired, the algorithm checks the *urgency strategy table*, executes the associated action and updates the status information.

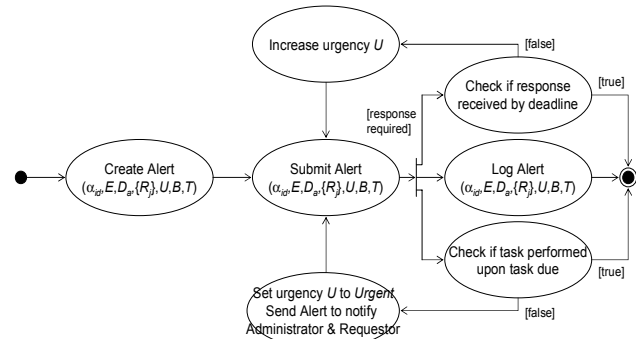


Figure 7. Control flow of the alerts in UML Activity Diagram

Figure 7 describes an overview of the flow control of alerts. The “Create Alert” node implements the creation of the alert as well as the value assignment (i.e., the execution of the matching algorithm). The “Submit alert” node sends the alert to the matched service provider (including humans such as doctors or nurses) and updates the alert to the active alerts table. The “Log Alert” node keeps the logging information.

The alert monitoring module is responsible for the “Check if response received by deadline” node as well as the “Increase urgency U ” node that results in the re-submission of an un-responded alert. The “Check if task performed upon task due” node is deadline triggered, i.e., if the associated task is not performed within deadline

(e.g., the doctor does not notify his arrival to the patient's location on time, or a patient record is not received within deadline), then it generates another alert. In addition, an exception alert is sent to the relevant administrator and the service requestor to notify this exception. As such, additional manual or system assisted exception handling processes [7] can be carried out by both parties.

4.5. Web Services Design for AMS

In this subsection, we enlist a selection of Web Services for the AMS as follows. An alert to a service provider can be requested through the Web service *requestAlert*. The requestor includes parameters encapsulation the requirements and description of the alert, together with optional specification of a deferred response channel (which defaults to a Web Service *receiveDeferredResponse*, and may be email, SMS, etc., for human requestors). In response, the service provider will send an acknowledgment to the requestor, indicating that the request is confirmed or denied, or the response will be deferred. Deferred responses and other exception alerts can be returned through the requestor's service *receiveDeferredResponse*. Further, the requestor may cancel the alert afterwards by calling the service *cancelAlert*.

Service Name: requestAlert

- *Input: AlertID, RequestorID, AlertMessage, Roles, Urgency, ResponseRequired (TRUE | FALSE), Deadline, Deferred Response Channel.*
- *Response: AlertID, ServiceProviderID, Ack (Confirmed | Denied | Deferred), ResponseMessage, AlertReceiptTime*

Service Name: cancelAlert

- *Input: AlertID, RequestorID*
- *Response: Ack (Confirmed | Denied | Deferred)*

Service Name: receiveDeferredResponse

- *Input: Item AlertID, ServiceProviderID, ResponseMessage, AlertReceiptTime*
- *Response: Ack (Confirmed, NotConfirmed)*

5. Implementation and Discussions

In this section, we demonstrate the applicability of the AMS with the medical house-call management application. We also discuss how overall E-service provision can be streamlined with this design.

5.1. Overall System Implementation Architecture

We have built a prototype on the J2EE and Oracle platforms [19]. Figure 8 depicts the overall implementation architecture of Medical House-Call System. As the AMS only manages the alert, domain-specific application logic

is required for a complete system. The application logic is triggered by the Process Execution Module of the AMS to carry out timely appropriate actions in response to corresponding alert events. In addition, the application logic supports the Web front-end and other requests from medical partners' Web Service requests, such as process status enquiries. In particular, we filter and validate requests and formulate them with alerts in our own format before feeding into the AMS. After that, the AMS can manage the alerts on its own, routing them to selected service providers (human or E-service providers) and monitoring the actions. Only in the case of exceptions as defined by the administrators (e.g., services are not performed within deadline), new alerts are generated to the administrative staff and the service requestors to trigger possible exception handling procedures (which may be pre-programmed or system supported manual activities).

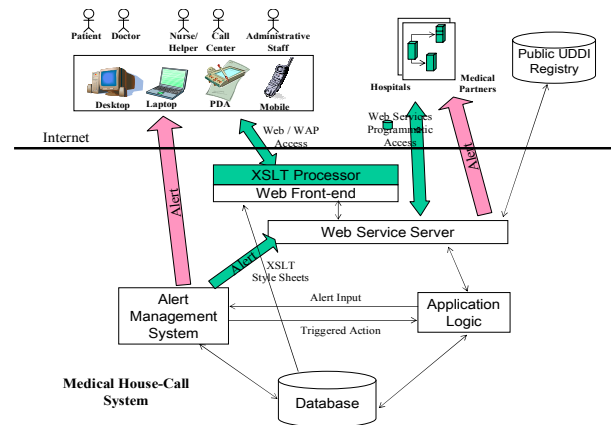


Figure 8. Overall implementation architecture of the Medical House-Call System



Figure 9. Sample alert acknowledge user interface for a doctor

To maintain maximum reusability of code, all external access to the application logic is to be performed through the Web Service interface. Even the Web front-end has to invoke these service points in order to maintain maximum code modularity. This approach is further justified because

the eXtended Message Language (XML) messages returned with Web Services can immediately be rendered with XML Stylesheet Language (XSL) technologies for users on different platforms. For example, different Hypertext Markup Language (HTML) outputs are generated for Web browsers on desktop PCs and PDAs respectively, while WAP Markup Language (WML) outputs are generated for mobile phones [15]. Figure 9 illustrates a sample alert acknowledgement interface for a doctor through WAP on a mobile phone.

5.2. System Walkthrough

There are some pre-operation for the system. First of all, the administrator has to define the urgency strategies as discussed in Section 4.3. In addition, the administrator has to specify the employees' roles and create employees' roster in order. After all the pre-operations are completed, the system is ready to receive request for house-calls.

When the call center obtains a house-call request from a user online or from a medical partner via Web Services, the AMS searches for a suitable doctor to take the call. The following task and alert is fed into the AMS: **Task** (T001, "Take a house-call (HC0001)", Doctor) and **Alert** (A001, Patient X, "Ask the doctors whether they can take a house call", {Doctor A, Doctor B, Doctor C}, Urgent, Y, T+20min). If a doctor is available for the house-call, he should reply with **Response** (R001, A001, Accept, 10May2003 13:05). If his is unavailable, he should reply with **Response** (R002, A001, Decline, 10May2003 13:10). According to the urgency of the alert, the AMS will choose a suitable doctor among the available doctors with *accept* replies based on an administrator defined cost function (see [6] for more details). The system will send notifications to confirm with the selected doctor and to cancel the request for other doctors, respectively. The selected doctor receives the following alert: **Result** (A002, Patient X, "You are confirmed to take the house-call.", Doctor A, Urgent, Y, T+10min). Other doctors receive the following alert: **Result** (A003, Patient X, "The house-call request is cancelled.", {Doctor B, Doctor C}, Normal, N, No deadline). Finally, the requestor receives the following response: **Result** (A004, Doctor A, "Your house-call request is confirmed by Doctor A", Patient X, Normal, No deadline)

In addition, the patient or the call-center operators may select their preferred doctor(s) with the specified specialties through a Web interface as shown in Figure 10. If so, the AMS will first consider the selected doctors first before other doctors.

As soon as the doctor is confirmed for the service, the system sends alerts to relevant hospitals and clinics for retrieving the patient's medical records. For privacy concern, these records are directly sent to the doctor's mobile device (and to the patient's personal device if possible and

required). The AMS monitors for the reception acknowledgement within a deadline, otherwise the AMS will raise an exception to draw an administrator's attention for manual follow-up. On the other hand, if the doctor does not acknowledge his arrival on time, the AMS will also raise an exception. The administrator then calls the doctor's mobile phone to estimate if a replacement is required. If the doctor actively cancels the assignment, the AMS can automatically look for another replacement doctor.

Figure 10. Sample User Interface for doctor selection

Call ID	Patient ID	Doctor	Admin Staff	Start Time	Status
HC0384	PH002993	N/A	N/A	11 May 2003 12:30	Finding Doctor, waiting for doctors' reply
HC3740	PH002992	Dr. Philip Ng	Terence Yeung	11 May 2003 10:05	On the way to local patient
HC1283	PH048737	Dr. Joanne Wong	Cindy Wong	11 May 2003 09:25	Consultation in progress
HC6483	PH009938	Dr. Steven Ip	Cindy Wong	11 May 2003 05:45	Replacement for absence of doctor
HC4588	PH006744	Dr. Amy Chan	May Cheung	11 May 2003 03:15	On the way to overseas patient
HC5448	PH005544	Dr. Gary Lee	Cindy Wong	11 May 2003 01:10	Consultation in progress, need extra help
HC2334	PH006222	Dr. Paul Yip	Gillian Chan	10 May 2003 23:55	Wait for Payment

Figure 11. House-call status monitor

For E-service provision, a vital administrative function is to monitor the status of service progress, and especially exceptions. Thus, the AMS generates alerts to relevant administrator(s) upon exception. The administrator can monitor house-call status through the House-call Status Monitor page (cf. Figure 11), in which manual manipulations can be carried out if necessary.

5.3. Discussion

This case study demonstrates that urgency requirements and the associated interactions with service partners (both human and E-service providers) can be systematically and modularly captured into an AMS. It should be noted that an AMS targets for urgent, asynchronous, unstructured, or even ad-hoc tasks (such as exception handling). Therefore, it is complimentary to conventional workflow management systems (WFMS) which targets at regular synchronous task flows. In fact, the motivation of

AMS evolves from the exception handling and user-interface mechanisms of our ME-ADOME WFMS [2], by factoring out and extending such functions (in particular urgency requirements). The physical execution of individual tasks is outside the scope of the AMS and is in capture in the application logic of individual information systems (as illustrated in Figure 8), which can well be a WFMS.

Such an AMS is light-weight and highly coherent, but loosely coupled with other sub-systems, enabling it to be plugged into any information system that needs such functionalities. Besides routing alerts to external service providers, an AMS can as well route alerts to other AMSs of a large enterprise. As such, AMSs can be physically distributed within an enterprise and into required systems. They are orchestrated by Web Services technology to work together seamlessly in the enterprise and even across organization boundaries in partner E-service providers. This architecture is highly scalable and interoperable. Further, there are no practical limitations in the implementation platform for each of these systems as long as they support Web Services and programmed to complied with common call conventions. For example, existing Java-2 Enterprise Edition (J2EE) enterprise applications can employ Sun Microsystems' Web Service solutions while current Microsoft PC-based systems may be extended with the .NET framework [18]. For legacy systems, wrappers may be built around them to enable compatibility with Web Services. As such, upgraded sub-systems can provide alert support through an AMS gradually for adequate testing and streamlining the switch-over, which may otherwise cause great impacts for major enterprise-wide systems.

In addition, Web Services serve as the middleware for interactions among business partners for alert-driven E-services in both directions. Similar gradual migration strategies are also possible. In order to further streamline interactions among enterprises, application layer semantics (such as content taxonomy and category definitions), protocols for interaction, and service-level standards are called for. Trade unions and regulatory bodies may help in such standardizations. If so, content service grids [11] can be formed for seamless and effective E-services enactment and monitoring can be realized.

We further consider the applicability of our AMS implementation framework to other industries is highly positive and optimistic. This is because of the trend that organizations are moving towards service-oriented operations. For large enterprises, our operation model for AMS based on the anticipated workflow is highly generic. Though some enterprises currently do not consider entertaining alert-driven E-services from clients, they eventually need to do so in order to increase their competitiveness. For smaller enterprises, it is feasible to plug in such a light-weighted AMS (as compared with a full-blown

WFMS) into their systems as software houses may develop packages with our approach.

6. Conclusions

The wide-spread of Internet connectivity and Web Services gradually changes the way in accessing and use information. Organizations are shifting towards an inter-operation paradigm in which quality and timely services are essential. Awareness, accessibility, and responsiveness are the key relationships among organizations in society.

In this paper, we looked into the problem efficiently conveying alerts to the right service provider at the right time using Web Services and mobile devices, for E-service provision under urgency constraints. We have proposed a framework of an alert management system that supports both human and E-service providers. This framework introduces a flexible alert conceptual model that allows users to specify tasks, alerts, roles, and their inter-relations. We have also presented our AMS architecture with an implementation outline with Web Services and described the alert monitoring and routing mechanisms involved. The problem of finding the right service provider to send an alert is non-trivial and application dependent, but our approach provides administrators with a guideline to systematically define strategies to address this problem. We have further proposed a matching algorithm based on our alert conceptual model that can be customized based on the application requirements. Moreover, we present an alert monitoring algorithm that monitors the alerts and raises their urgency accordingly while at the same time executes the action that corresponds to the according urgency level. As such, an effective AMS can be built and it can be reused by plugging into systems that require sophisticated alert support. We have demonstrated the applicability of the AMS in a medical house-call system, which need to manage timely services from both human professionals and E-service providers. With this approach, E-service grids can gradually be built up among a group of E-service providers.

We are also implementing the AMS under our ME-ADOME environment [6], aiming to strengthen the support for alerts for general workflow and E-service management. We are investigating in inter-relations among alerts. In particular, we are looking into alerts due to failure of commitments [3], their relation to contract enforcement, and in workflow based information integration. We are also interested in further issues of adaptation for mobile application [3], the impact of cancellations, and other possible exceptions. We are also interested in the tradeoff between quality and cost, and service negotiation.

References

- [1] E. Ammenwerth, A. Buchauer, B. Bludau, and R. Haux, "Mobile information and communication tools in the hospital," *Intl. J. of Medical Informatics*, 57 (1):21-40., 2000.
- [2] D.K.W. Chiu, S.C. Cheung, and E. Kafeza, "Three-tier View-based Support for Mobile Workflow," In *Proc. 1st Intl. Conf. on Mobile Business*, CDROM, Athens, Greece, July 2002.
- [3] D.K.W. Chiu, S.C. Cheung, E. Kafeza, and H.F. Leung, "A Three-Tier View Methodology for adapting M-services," *IEEE TSMC, Part A*, 2004, (to appear).
- [4] D.K.W. Chiu, S.C. Cheung, K. Karlapalem, Q. Li, Sven Till, and E. Kafeza, "Workflow View Driven Cross-Organizational Interoperability in a Web Service Environment," *Information Technology and Management*, 2004 (to appear).
- [5] D.K.W. Chiu, S.C. Cheung, and S. Till, "An Architecture for E-Contract Enforcement in an E-service Environment," In *Proc. HICSS36*, CDROM, Jan 2003.
- [6] D.K.W. Chiu, Q. Li, and K. Karlapalem, "A Meta Modeling Approach for Workflow Management System Supporting Exception Handling," *Information Systems*, 24(2):159-184, 1999.
- [7] D.K.W. Chiu, Q. Li, and K. Karlapalem, "Web Interface-Driven Cooperative Exception Handling in ADOME Workflow Management System," *Information Systems*, 26(2):93-120, 2001.
- [8] D.K.W. Chiu, K. Karlapalem, Q. Li, and E. Kafeza, "Workflow Views Based E-Contracts in a Cross-Organization E-Service Environment," *Distributed and Parallel Databases*, 12(2-3):193-216, 2002.
- [9] S. Eisenstadt, M. Wagner, W. Hogan, M. Pankaskie, and F-C Tsui. W. Wilbright, "Mobile workers in healthcare and their information needs: are 2-way pagers the answer?" In *Proc. AMIA Annual Fall Symposium*, 1998, pp.135-139.
- [10] A. Fano and A. Gershman, "The Future of Business Services in the age of Ubiquitous Computing," *CACM*, 45(12):83-87, 2002
- [11] W. Gentzsch. Grid computing: a new technology for the advanced web. In *Proceedings of the NATO Advanced Research Workshop on Advanced Environments, Tools, and Applications for Cluster Computing*, Lecture Notes in Computer Science, Springer-Verlag, 2326:1-15, 2002.
- [12] I. Haimowitz, J. Farley, G.S. Fields, J. Stillman, and B. Vivier, "Temporal Reasoning for automated workflow in Helath Care Enterprises," *Electronic Commerce: Current Reasearch Issues and Applications*, LNCS 1028, Springer, 1996, pp87-113
- [13] G.. Hripcsak, P. Clayton, R.A. Jenders, J.J. Cimino, and S.B. Johnson, "Design of a Clinical Event Monitor," *Computers and Biomedical Research*, 29:194-221, 1996.
- [14] P. Hung and D.K.W. Chiu, "Developing Workflow-based Information Integration with Exception Support in a Web Services Environment," In *Proc. HICSS37*, Jan 2004, CDROM, 10 pages.
- [15] Y.-B. Lin and I. Chlamtac, *Wireless and Mobile Network Architectures*. John Wiley & Sons, 2000.
- [16] F. Malamateniou and G. Vassilacopulos, "A Workflow-based Approach to Virtual Patient Record Security," *IEEE TITB*, 2(3):39-145, 1998.
- [17] Object Management Group. *Foreword UML specification 1.4*, September 2001. <http://www.omg.org/>
- [18] D. Peltzer. *.NET & J2EE Interoperability*, McGraw-Hill Osborne, 2003.
- [19] J. Price. *Oracle 9i JDBC Programming*, McGraw-Hill Osborne, 2002.
- [20] D.M. Ride, C. Safran, R.S. Philips, Q. Wang, D.R. Calkins, T.L. Delbanco, H.L. Bleich, and W.V. Slack, "Effect of Computer Based Alerts on the Treatment and Outcomes of Hospitalized Patients," *Archives of Internal Medicine*, 154:1511-1517, 1994.
- [21] W. Raghupathi and J. Tan, "Strategic IT Applications in health care," *CACM*, 45(12)56-61, 2002.
- [22] P. Weverka. *Mastering ICQ: The Official Guide*. IDG Books. ICQ Press, 2000, <http://www.icq.com>
- [23] Workflow Management Coalition. *The Workflow Reference Model*. (WFMC-TC-1003, 19-Jan-95, 1.1)